Matan Suliman

Drone Precision Landing on a Moving Platform

Workshop in Autonomous Systems Simulation (Fall 2025)

Today's talk

Requirements

Environmental constraints

System architecture

Sequence Diagram

Class Diagram

Scenario

Evaluation criteria

Live Run

Future Design

Alternative Design

Requirements

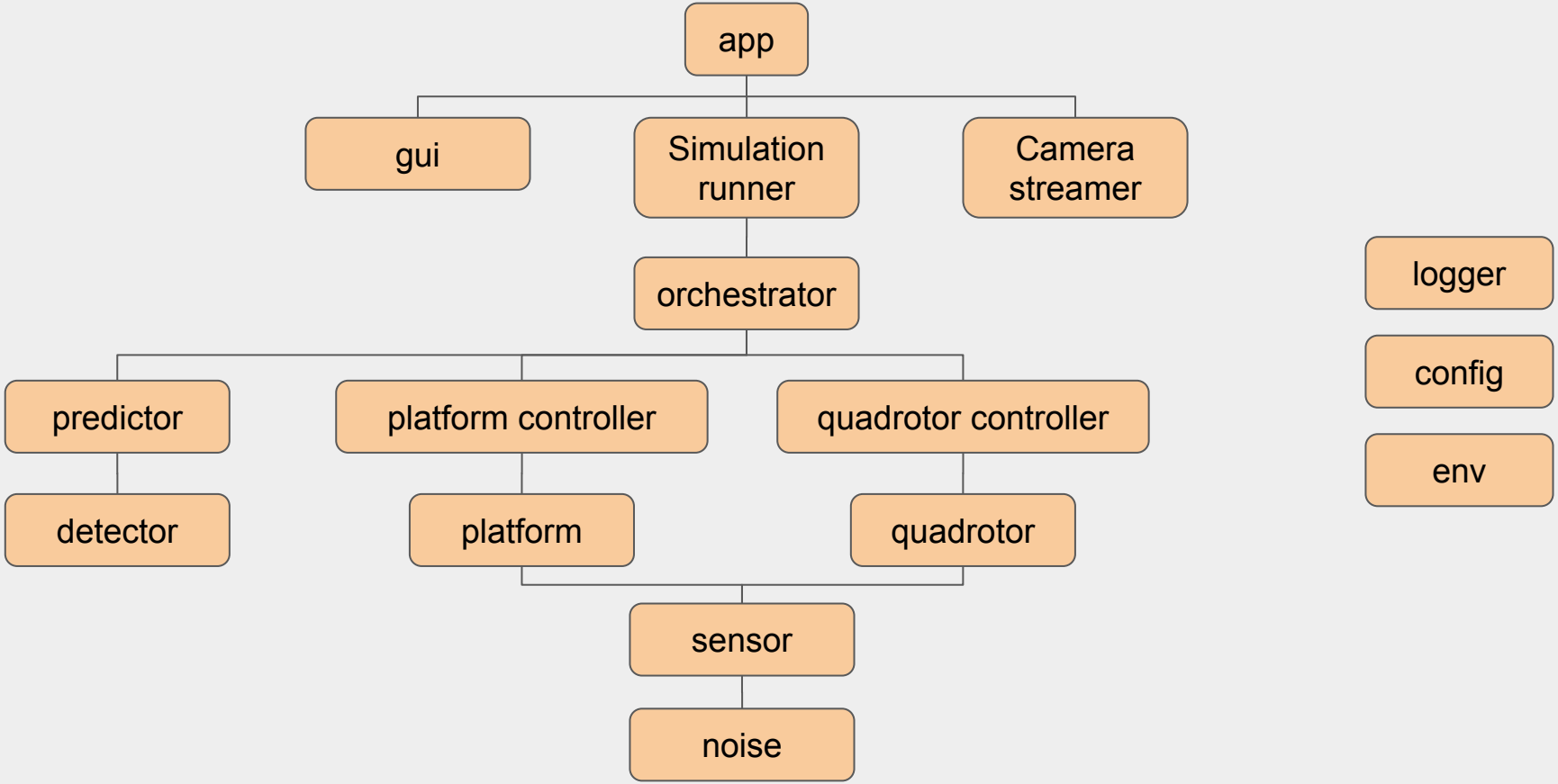
- Autonomous guidance
- Real time computation

Constraints

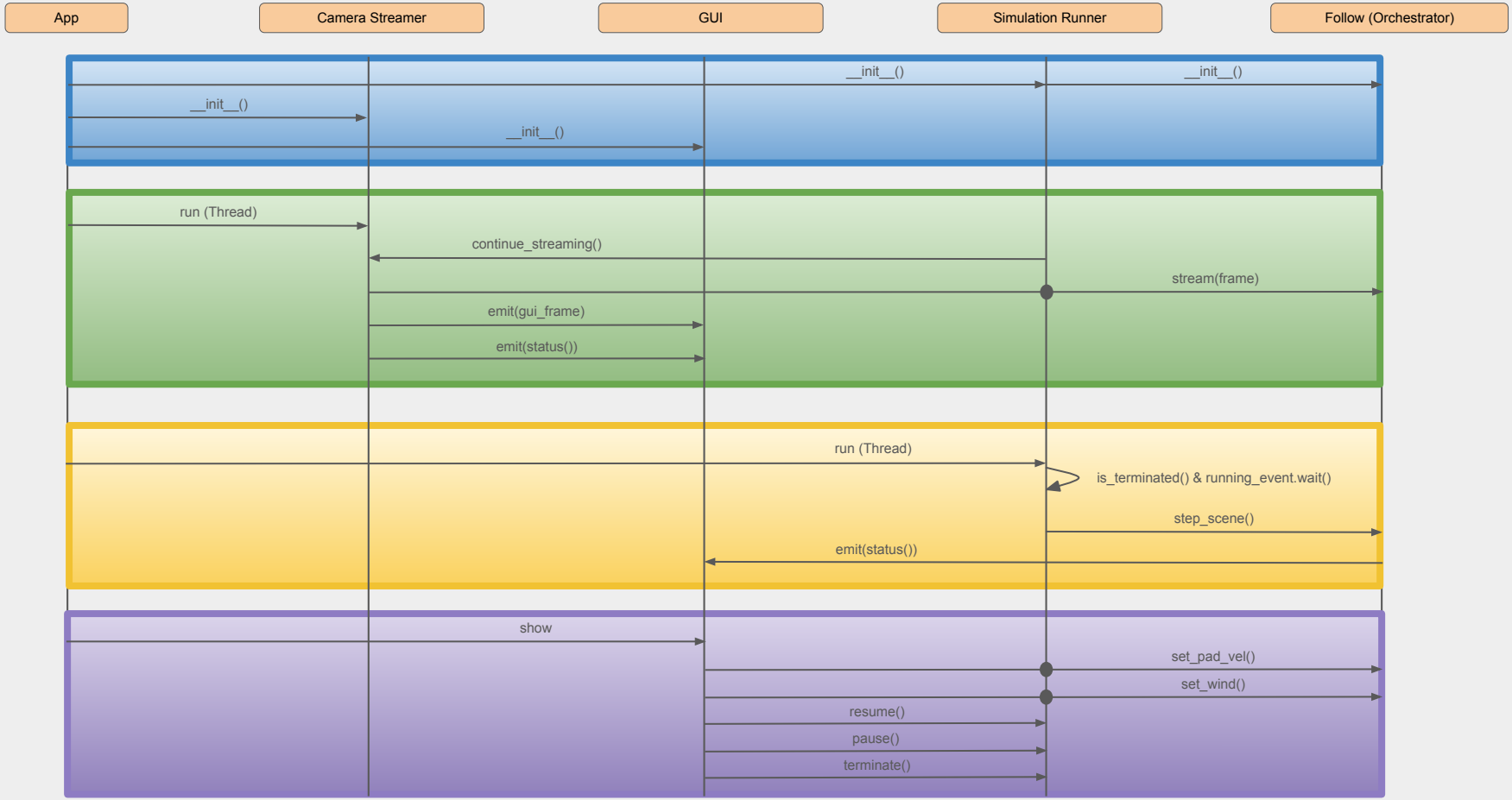
- No obstacles
- Platform: constant* vx and vy limited to +-5 m/s, vz = 0
- Camera: Limited resolution
- Wind: constant*, limited to +-10 m/s
- Sensors noise: constant and only in x,y axis of pos

*can be altered

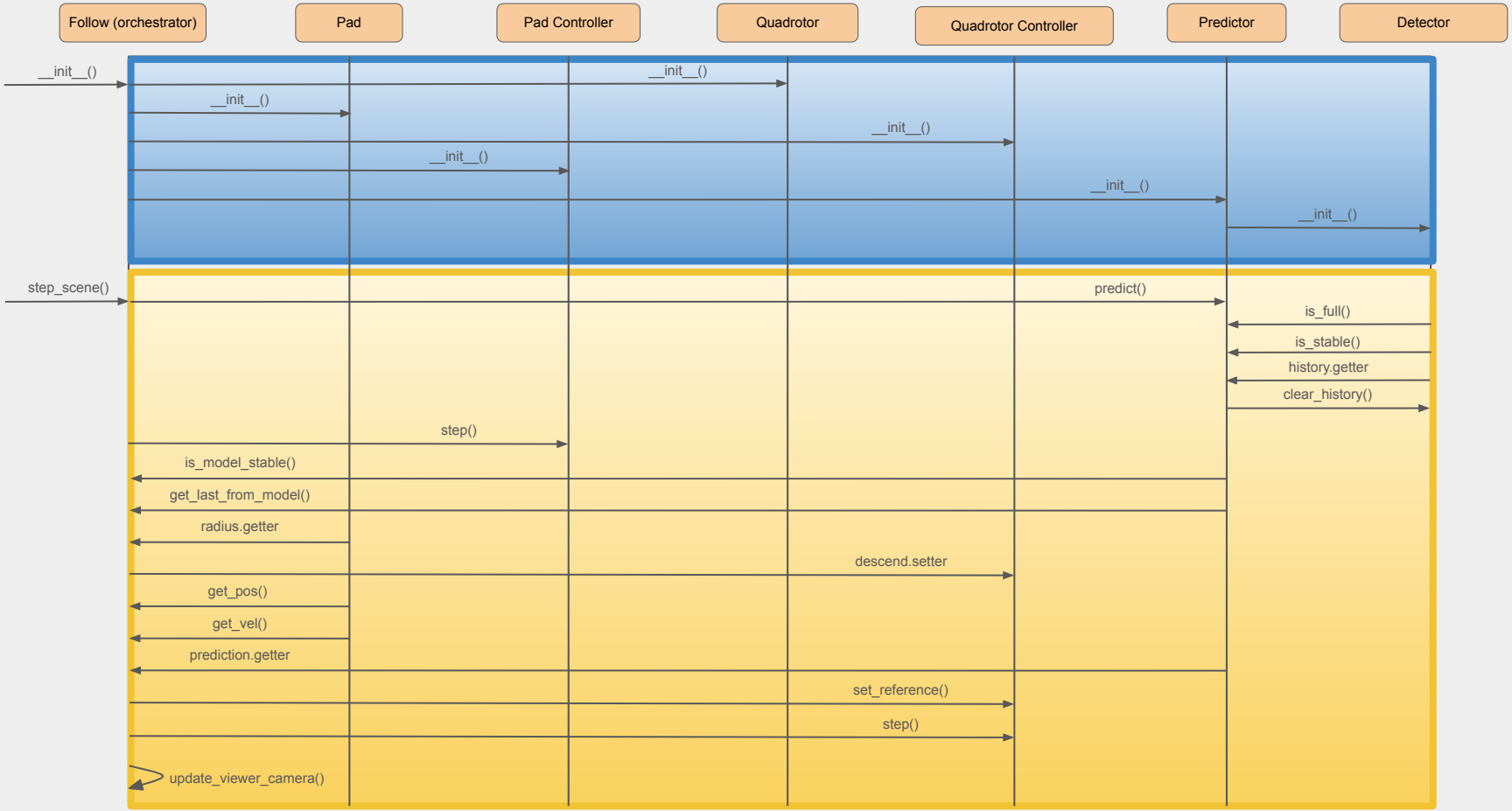
System architecture

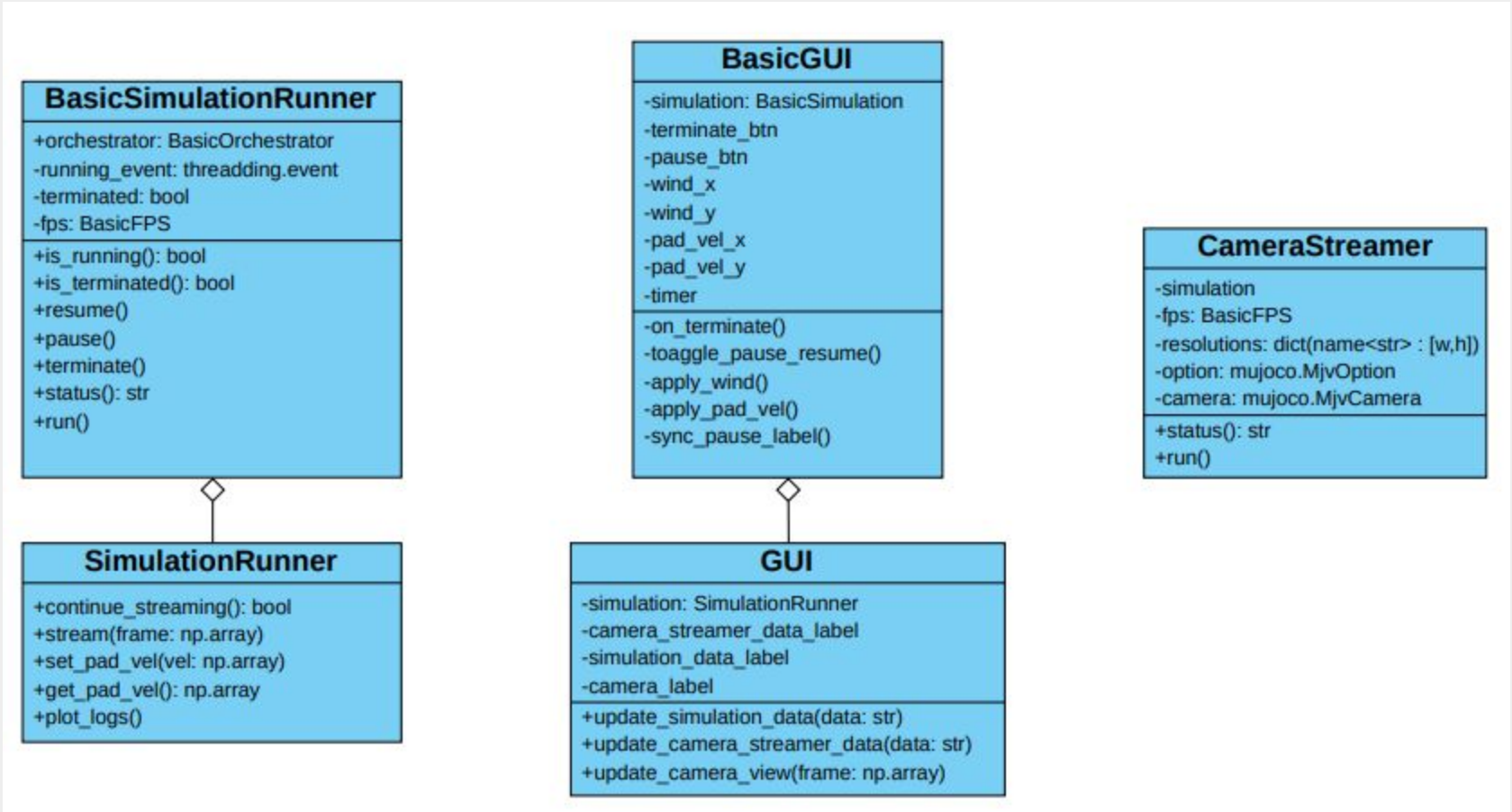


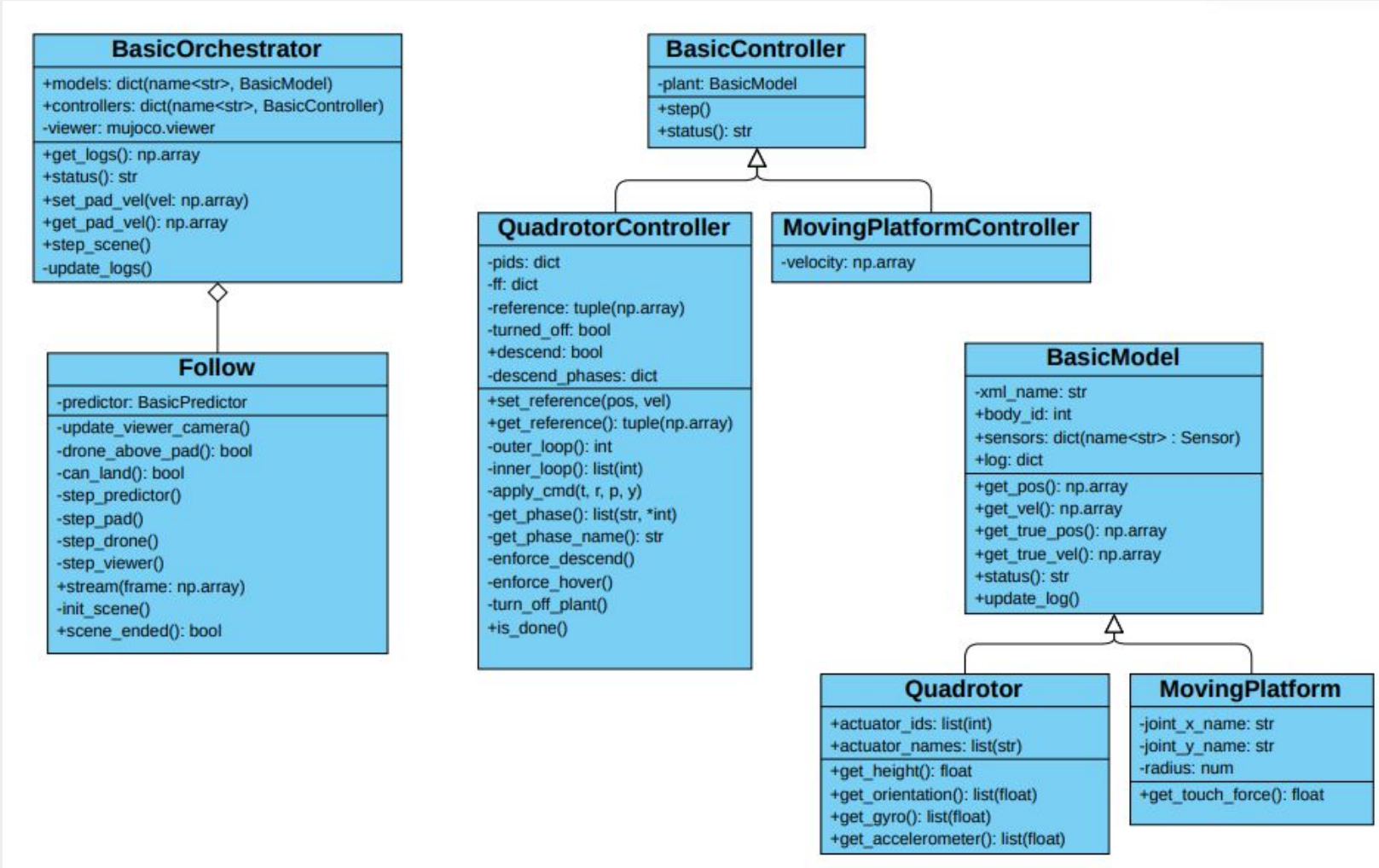
Sequence Diagram App Threads

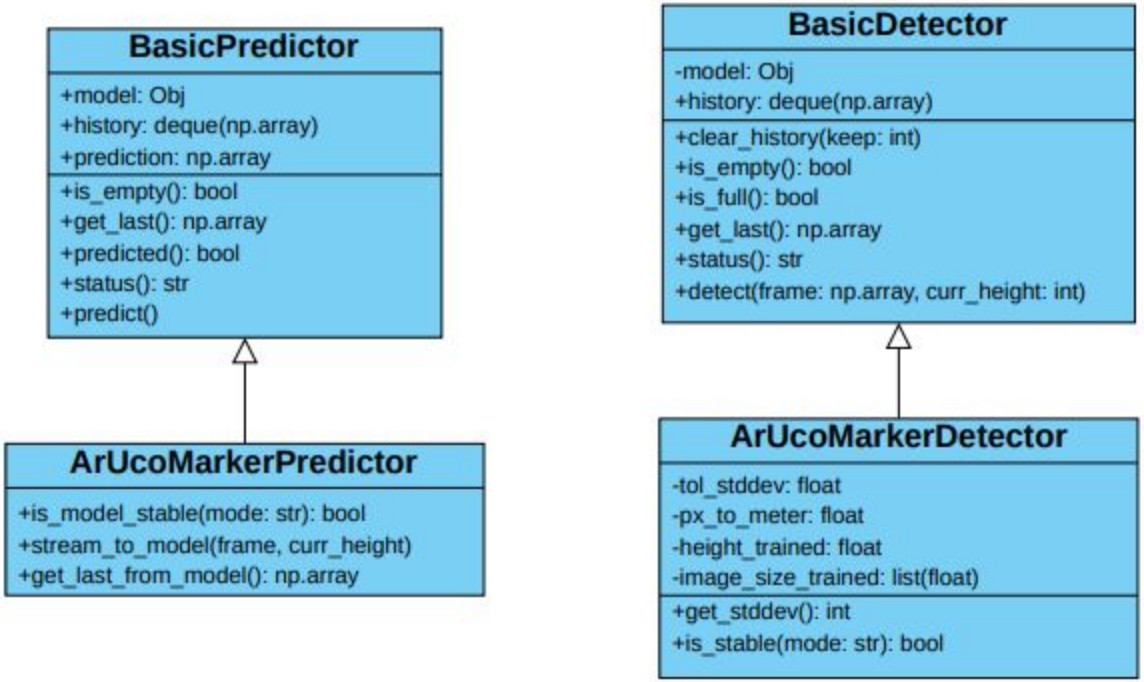


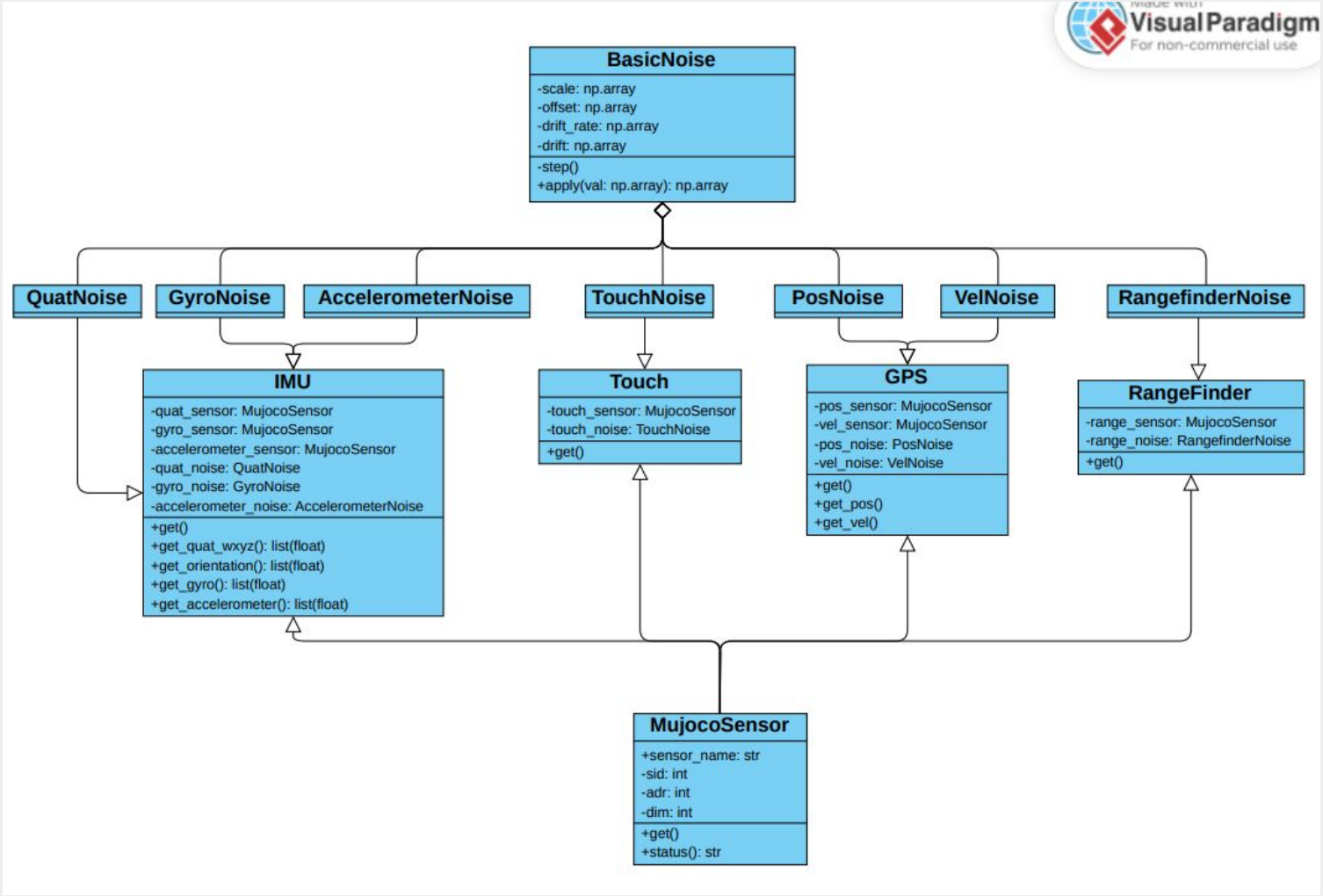
Sequence Diagram Orchestrator




BasicSimulationRunner
+orchestrator: BasicOrchestrator
-running_event: threadding.event
-terminated: bool
-fps: BasicFPS
+is_running(): bool
+is_terminated(): bool
+resume()
+pause()
+terminate()
+status(): str
+run()
SimulationRunner
+continue_streaming(): bool
+stream(frame: np.array)
+set_pad_vel(vel: np.array)
+get_pad_vel(): np.array
+plot_logs()
BasicGUI
-simulation: BasicSimulation
-terminate_btn
-pause_btn
-wind_x
-wind_y
-pad_vel_x
-pad_vel_y
-timer
-on_terminate()
-toaggle_pause_resume()
-apply_wind()
-apply_pad_vel()
-sync_pause_label()
GUI
-simulation: SimulationRunner
-camera_streamer_data_label
-simulation_data_label
-camera_label
+update_simulation_data(data: str)
+update_camera_streamer_data(data: str)
+update_camera_view(frame: np.array)
CameraStreamer
-simulation
-fps: BasicFPS
-resolutions: dict(name<str> : [w,h])
-option: mujoco.MjvOption
-camera: mujoco.MjvCamera
+status(): str
+run()


BasicOrchestrator
+models: dict(name<str>, BasicModel)
+controllers: dict(name<str>, BasicController)
-viewer: mujoco.viewer
+get_logs(): np.array
+status(): str
+set_pad_vel(vel: np.array)
+get_pad_vel(): np.array
+step_scene()
-update_logs()
Follow
-predictor: BasicPredictor
-update_viewer_camera()
-drone_above_pad(): bool
-can_land(): bool
-step_predictor()
-step_pad()
-step_drone()
-step_viewer()
+stream(frame: np.array)
-init_scene()
+scene_ended(): bool
BasicController
-plant: BasicModel
+step()
+status(): str
QuadrotorController
-pids: dict
-ff: dict
-reference: tuple(np.array)
-turned_off: bool
+descend: bool
-descend_phases: dict
+set_reference(pos, vel)
+get_reference(): tuple(np.array)
-outer_loop(): int
-inner_loop(): list(int)
-apply_cmd(t, r, p, y)
-get_phase(): list(str, *int)
-get_phase_name(): str
-enforce_descend()
-enforce_hover()
-turn_off_plant()
+is_done()
MovingPlatformController
-velocity: np.array
BasicModel
-xml_name: str
+body_id: int
+sensors: dict(name<str> : Sensor)
+log: dict
+get_pos(): np.array
+get_vel(): np.array
+get_true_pos(): np.array
+get_true_vel(): np.array
+status(): str
+update_log()
Quadrotor
+actuator_ids: list(int)
+actuator_names: list(str)
+get_height(): float
+get_orientation(): list(float)
+get_gyro(): list(float)
+get_accelerometer(): list(float)
MovingPlatform
-joint_x_name: str
-joint_y_name: str
-radius: num
+get_touch_force(): float


BasicPredictor
+model: Obj
+history: deque(np.array)
+prediction: np.array
+is_empty(): bool
+get_last(): np.array
+predicted(): bool
+status(): str
+predict()
ArUcoMarkerPredictor
+is_model_stable(mode: str): bool
+stream_to_model(frame, curr_height)
+get_last_from_model(): np.array
BasicDetector
-model: Obj
+history: deque(np.array)
+clear_history(keep: int)
+is_empty(): bool
+is_full(): bool
+get_last(): np.array
+status(): str
+detect(frame: np.array, curr_height: int)
ArUcoMarkerDetector
-tol_stddev: float
-px_to_meter: float
-height_trained: float
-image_size_trained: list(float)
+get_stddev(): int
+is_stable(mode: str): bool


BasicNoise
-scale: np.array
-offset: np.array
-drift_rate: np.array
-drift: np.array
-step()
+apply(val: np.array): np.array
QuatNoise
GyroNoise
AccelerometerNoise
TouchNoise
PosNoise
VelNoise
RangefinderNoise
IMU
-quat_sensor: MujocoSensor
-gyro_sensor: MujocoSensor
-accelerometer_sensor: MujocoSensor
-quat_noise: QuatNoise
-gyro_noise: GyroNoise
-accelerometer_noise: AccelerometerNoise
+get()
+get_quat_wxyz(): list(float)
+get_orientation(): list(float)
+get_gyro(): list(float)
+get_accelerometer(): list(float)
Touch
-touch_sensor: MujocoSensor
-touch_noise: TouchNoise
+get()
GPS
-pos_sensor: MujocoSensor
-vel_sensor: MujocoSensor
-pos_noise: PosNoise
-vel_noise: VelNoise
+get()
+get_pos()
+get_vel()
RangeFinder
-range_sensor: MujocoSensor
-range_noise: RangefinderNoise
+get()
MujocoSensor
+sensor_name: str
-sid: int
-adr: int
-dim: int
+get()
+status(): str

Scenario

The drone hovers and the platform start moving

follows and catches up to the platform

Bottom camera streams to marker detection model

Prediction model makes a prediction of the mean detections

The drone gets a better evaluation of the platform position

The drone goes through the landing phases

Evaluation

- Success or Fail : Land or Crash
- Accuracy: x,y absolute error
- Delay: time to land

Live Run

Future Design

- Tune hyper parameters from config file
- Platform controller: vz != 0, dynamic movement
- Wind model: dynamic, different types
- Sensors noise: dynamic, on sensors

New:

- Orchestrator: abort mission
- Fog model
- FPS model: better real time computations

Alternative Designs

change	**What to do update**
drone	drone.xml, models.py, controllers.py
platform	platform.xml, models.py, controllers.py
gui	gui.py
predictor	predictor.py
noise	noises.py
sensor	sensors.py
scenario	orchestrator.py, scene.xml